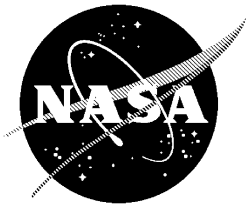
**CSOC MISSION SERVICES**

# Scalable, Integrated, Multimission Simulation Suite (SIMSS) Functional and Performance Requirements

**September 2001**




National Aeronautics and Space Administration

Goddard Space Flight Center
Greenbelt, Maryland

# Scalable, Integrated, Multimission Simulation Suite (SIMSS)
# Functional and Performance Requirements

**September 2001**

Prepared Under Contract NAS9-98100
CSOC Task Order SODA GM-03
By Computer Sciences Corporation

**Prepared by:**

P. Yeh Date
Technical Lead, SIMSS, CSC

**Reviewed by:**

P. Yeh Date
System Test Lead, SIMSS, CSC

K. Lavery Date
Independent Test Lead, SOC, Honeywell-TSI

**Quality Assured by:**

L. Hepfer Date
Quality Management Office, CSC

**Approved by:**

D. Green Date
Task Lead, SIMSS, CSC

E. Noone Date
CSOC Customer Service Representative, SIMSS, CSC

E. Chandler Date
Task Manager, SIMSS
GSFC, Code 583

**Goddard Space Flight Center**
Greenbelt, Maryland

# Preface

This document contains the functional and performance requirements for the Scalable, Integrated, Multimission Simulation Suite (SIMSS). This functional and performance specification, once baselined will be controlled by the SIMSS Configuration Control Board.

This functional and performance specification was prepared by

SIMSS Project
Code 583
Goddard Space Flight Center
Greenbelt, Maryland 20771

Contributors to this document include the following:

Neil Ottenstein

Kevin Parker

Ernest Quintin

Jean Carlson

John Kim

Joe Polesel

Dave Green

Estelle Noone

Bill Parlock

Matt McCoy

Por-Wun Yeh

Chris Cohoon

Phong Tran

The changes made for Version 1.4 (SIMSS Release 5) of this document are summarized as follows:

- Added CG2 to CG3.5, and the entire DU subsection.
- Modified TM1 and TM4 Requirements; Changed TM2 and TM3 Comments.
- Changed IP Mode Performance 3.3.2 and 3.3.3 Status and Comments.
- Modified TG1.1.3, TG1.2.3, TG1.3.3; Changed the Status of TG1.1.5 to Partial and TG1.2.1, TG1.2.4, TG1.2.5, TG1.3.2, TG1.4, and TG3.4.4 to Full; Added TG1.2.6, TG1.2.7, and TG1.6; Changed the Comments for TG1.1.4 and TG1.3.4.
- Changed the Status of MD5.2 to Full.
- Changed Status for UI9.2, UI10, DT5, DT5.3, DT5.4, DT13.4, CI2.8 to Full. Changed Requirements and Comment for DM1 and DM2. Added Comment for DT13.5 and CI2.8. Deleted CI2.5 item f, CG1.6.3, CG1.6.4, CG1.6.5

# Abstract

The document presents the functional and performance requirements for the Scalable, Integrated, Multimission Simulation Suite (SIMSS). The SIMSS is being developed by the Goddard Space Flight Center (GSFC) of the National Aeronautics and Space Administration (NASA).

*Keywords: Scalable, Integrated, Multimission Simulation Suite (SIMSS); simulator; spacecraft; telemetry; commanding; communications; network; test tool; ground system; real-time*

# Change Information Page

| List of Effective Pages | |
|---|---|
| **Page Number** | **Issue** |
| Title Page | 1.4 |
| Front Matter | 1.2 |
| Section 1 | 1.2 |
| Section 2 | 1.4 |
| Section 3 | 1.4 |
| Abbreviations and Acronyms | 1.1 |

| Document History | | | |
|---|---|---|---|
| **Document Number** | **Status/Issue** | **Publication Date** | **CCR Number** |
| N/A | Original (Ver. 1.0) | April 2000 | N/A |
| N/A | Ver. 1.1 | January 2001 | 1 (R3 updates) |
| N/A | Ver. 1.2 | April 2001 | 2 (R4.1 updates) |
| N/A | Ver. 1.3 | May 2001 | 3 (R4.2 updates) |
| N/A | Ver. 1.4 | September 2001 | 4 (R5.0 updates) |

# Contents

# Section 1. Introduction

## 1.1 Purpose

This document sets forth the highest level functional and performance requirements (F&PR) for the Scalable, Integrated, Multimission Simulation Suite (SIMSS). These constitute the level-3 requirements for the SIMSS.

SIMSS is a both an infrastructure and a set of components that work within the infrastructure, which together provide an integrated collection of simulation, data generation, and data analysis tools needed for spacecraft ground systems testing and training.

## 1.2 Background

The goal of SIMSS is to reduce the cost of space operations in the related areas of spacecraft simulations and ground system test tools. This is to be accomplished through:

- A highly modular, open, "plug-and-play" software architecture
- Development of a baseline software platform with clearly defined interfaces and containing functionality common to all components
- Development of reusable software components using object-oriented technology
- Inexpensive, general-purpose hardware platforms
- State-of-the-art development tools and methodology
- Ongoing investigations into the latest technology for data communications, component-based development, real-time systems, and modeling, and incorporation of the new technology into the baseline system if found to be advantageous

## 1.3 Document Organization

This document is composed of three sections. Section 1 contains the document introduction and an overview of the SIMSS. It presents a brief summary of the scope, capabilities, and operations approach. Section 2 contains the functional requirements at the general system level and for each test configuration level. Section 3 contains the system-level performance requirements and the performance requirements for each test configuration. An abbreviations and acronyms list is provided for reference.

## 1.4 SIMSS Overview

The subsections that follow provide an overview of SIMSS. SIMSS is a component-based system within a client-server architecture. The graphical user interface (GUI) associated with each of the following modules represents the client portion of the architecture. The data processing portions of the modules represent the server side of the architecture. The SIMSS includes modules for:

- Receiving IP data
- Transmitting IP data

- Receiving serial data
- Transmitting serial data
- Logging data to a file
- Playing back data from a file
- Adding headers to data blocks
- Removing and validating headers from data blocks
- Monitoring data quality
- Encoding and decoding data
- Generating a telecommand stream based on user configuration
- Validating a telecommand stream based on user configuration
- Generating a telemetry stream based on user configuration
- Validating a telemetry stream based on user configuration
- Simulating a spacecraft (including telecommand validation, telemetry generation, reflection of commands in telemetry, and a scalable modeling capability) based on user configuration

All of these modules are expected to be reused in many test configurations. In addition, SIMSS provides a framework for quickly developing project-specific systems, such as spacecraft or ground station simulators, using generic components tailored to provide specific capabilities.

### 1.4.1 Scope

SIMSS is intended to be widely useful throughout spacecraft ground communication systems. Capabilities include but are not limited to:

- Generation of a data stream in an operator-configurable format in order to test communications interfaces
- Conversion of a data stream from one format to another in order to facilitate data transportation and to allow communications between otherwise incompatible systems
- Data quality checking on a data stream in order to verify communications interfaces
- Logging of a data stream for offline checking in order to verify communications
- Creating a data stream from playing back a file in order to test communications interfaces
- Generating telecommands in order to test a spacecraft or spacecraft simulator
- Validating telecommands in order to test a spacecraft control center or the communications between a control center and another location
- Generating telemetry in order to test a spacecraft control center or the communications between a control center and another location

- Simulating a spacecraft in order to provide communications testing, control center testing, or flight operations team (FOT) training

### 1.4.2 Test Capabilities

SIMSS will be capable of simulating any component of a spacecraft ground communications system for the purpose of communications or component testing or operator training.

### 1.4.3 Operations Overview

SIMSS will providing operations capabilities in several modes, depending on user needs and the time available.

At the simplest level, SIMSS is dynamically configurable and capable of being modified in realtime to generate, manipulate, validate, or record one or more datastreams for the purpose of communications testing.

Additionally, any SIMSS configuration is capable of being saved and reused at a later time.

Sophisticated modeling, simulation, or data generation capabilities will require advance preparation. However, SIMSS is designed to be data-driven, i.e., configurable based on information in a database and in flat files. So the user is given the capability to configure the system by modifying these files accordingly.

# Section 2. Functional Requirements

This section contains the functional requirements for the SIMSS.

## 2.1 Architectural Requirements

| Req. No. | Requirement | Release 5.0 Impl. | Comments |
|---|---|---|---|
| ***AR*** | ***Architectural Requirements*** | ***R5.0 Impl.*** | ***Comments*** |
| AR1 | The SIMSS shall have a client/server architecture with the client providing the user interface and the server providing the functionality. | Full | |
| AR2 | The SIMSS shall be capable of using multiple clients and servers. | Partial | |
| AR2.1 | The SIMSS shall be capable of interfacing with multiple servers from a single client. | Full | |
| AR2.2 | The SIMSS shall be capable of interfacing with multiple clients from a single server. Only one client shall be allowed to control the server with the other clients providing a display-only capability. | Partial | Allows multiple projects with multiple clients; but not single project with multiple clients. Also, all of the clients can control server. |
| AR2.3 | The SIMSS shall be capable of providing password protection for invoking or transferring the master user interface. | | |
| AR3 | The SIMSS shall be able to run client and server on same or separate hosts. | Full | |
| AR4 | Any SIMSS client shall be capable of running with any SIMSS server. | Full | |
| AR5 | The SIMSS client shall be capable of running on any pure Java compliant virtual machine. | Full | |
| AR6 | The SIMSS server shall be capable of running on a machine with a Windows NT operating system. | Full | |
| AR7 | The SIMSS server shall be capable of running on a machine with a Linux operating system. | | |
| AR8 | The SIMSS shall provide remote access to the SIMSS server | Full | |

| | | | |
|---|---|---|---|
| | for operations. | | |
| AR9 | The SIMSS shall use TBD security standards for remote access. | | |
| AR10 | The SIMSS shall consist of a collection of independent modules capable of being connected together via links for a specific function. | Full | |
| AR10.1 | The modules shall provide a standard user and programmatic interface. | Full | |
| AR10.2 | Each module shall have a client and a server component. | Full | |
| AR10.3 | Each module shall provide a specific, logically distinct element of overall SIMSS functionality. | Full | |
| AR11 | The SIMSS shall be capable of creating and running one or more configurations (projects) under operator control. A project is a collection of SIMSS modules and links intended to perform a specific function. | Full | |
| AR12 | The output channel of any SIMSS module shall be able to interface with the input channel of any other SIMSS module, and vice versa. | Partial | See limitations (Attachment H of R5.0 delivery package) |
| AR13 | The SIMSS shall be capable of being a component of a larger simulation system that is IEEE-1516 (DMSO HLA) compliant. | | |

## 2.2 User Interface Requirements

| *UI* | *User Interface Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| UI1 | The SIMSS shall provide a graphical user interface. | Full | |
| UI1.1 | The SIMSS shall provide a user interface that can be run within a Web browser. | | |
| UI1.2 | The SIMSS shall provide a text command line interface for directives to be entered. | Full | |
| UI1.3 | The SIMSS shall provide the same degree of control from directives that is available from the graphical user interface. | Partial | Not all modules |
| UI1.4 | The SIMSS shall be capable of generating a scenario file containing all the directives entered by the user during a specific period of time. | | |
| UI1.5 | The SIMSS shall provide the capability to recall previously entered directives to be executed again or edited and then executed up to at least the last ten directives entered | Full | |
| UI2 | The SIMSS shall provide the user with project control. | Full | |

| | | | |
|---|---|---|---|
| UI2.1 | The SIMSS shall provide the user with the capability to create and delete projects. | Full | |
| UI2.2 | The SIMSS shall provide the user with the capability to select which server host to connect to for a specific project. | Full | |
| UI2.3 | The SIMSS shall provide the user with the capability to add a module to a project. | Full | |
| UI2.4 | The SIMSS shall provide the user with the capability to delete a module from a project. | Full | |
| UI3 | The SIMSS shall provide the user with module channel control. | Full | |
| UI3.1 | The SIMSS shall provide the user with the capability to determine how many input and output channels a module is capable of handling. | Full | |
| UI3.2 | The SIMSS shall provide the user with the capability to create a link between an output channel of any module to an input channel of any other module. | Full | |
| UI3.3 | The SIMSS shall provide the user with the capability to delete any link previously created. | Full | |
| UI3.4 | The SIMSS shall provide the user with the capability to determine which channels a link connects. | Full | |
| UI3.5 | The SIMSS shall be capable of displaying meaningful names for each module input or output channel. | | |
| UI4 | The SIMSS shall be capable of displaying a brief description of a module's function. | Full | |
| UI5 | The SIMSS shall provide the user with the capability to stop and start operations for a specific project. | Full | |
| UI6 | The SIMSS user interface shall be capable of displaying and logging all system event messages for a specific project. | Full | |
| UI6.1 | The SIMSS user interface shall log all system messages for a specific project to an event log that is accessible offline if SIMSS is not running. | Full | |
| UI6.2 | The SIMSS user interface shall provide a display showing all system messages generated during the current session for a specific project. | Full | |
| UI6.3 | The SIMSS user interface shall provide the capability to filter system messages based on message type. | Full | |
| UI7 | The SIMSS user interface shall be capable of printing any display. | Full | Copy/paste to/from clipboard |
| UI8 | The SIMSS shall provide the user with the capability to save the configuration of a current project. | Partial | |
| UI8.1 | The SIMSS shall be capable of saving the overall configuration (modules and links) of a project. | Full | |
| UI8.2 | The SIMSS shall be capable of saving the configuration information specific and internal to modules in a project. | Partial | Not all modules |

| | | | |
|---|---|---|---|
| UI9 | The SIMSS shall provide the user with the capability to restore a project based on a previously stored configuration. | Full | |
| UI9.1 | The SIMSS shall be capable of restoring the overall configuration (modules and links) of a project based on a saved configuration | Full | |
| UI9.2 | The SIMSS shall be capable of restoring the configuration information specific and internal to modules in a project based on a saved configuration. | Full | For all modules that need save/restore capability. |
| UI10 | Each SIMSS module shall provide the user with the capability to control its configuration and functionality. | Full | |
| UI11 | Each SIMSS module shall provide the user with the capability to monitor its configuration and status. | Full | |
| UI12 | The SIMSS user interface shall update all dynamic displays at least once every three seconds. | Full | |
| UI13 | Each SIMSS module shall be capable of displaying to the user the date and release number of the version currently running. | Partial | It may not reflect the latest date of changes if the *DllMainClass .cpp was not checked in at the same time. |

## 2.3 Data Management Requirements

| *DM* | *Data Management Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| DM1 | The SIMSS User's Guide (SUG) shall define a standard storage format and medium for all information needed to generate and modify telemetry, validate and identify commands, and reflect commands in telemetry. | Partial | TDM telemetry and commands, and CCSDS telemetry have been implement-ed. |
| DM2 | All SIMSS modules that generate or modify telemetry, validate or identify commands, or reflect commands in telemetry shall adhere to the formats described in SUG when storing and retrieving data used for these purposes. | Partial | See DM1. |
| DM3 | The SUG shall include record formats for information about each of the following telemetry elements:<br>a. Telemetry parameters<br>b. Telemetry locations | Partial | a,b,c,e,f,g,h only. |

| | | | |
|---|---|---|---|
| | c. Telemetry packets<br>d. TDM telemetry formats<br>e. Physical channels<br>f. Virtual channels<br>g. Virtual channel to physical channel mappings<br>h. Packet to virtual channel mappings<br>i. Polynomial conversions between raw telemetry values and engineering units<br>j. Linear conversions between raw telemetry values and engineering units<br>k. Conversions between raw telemetry values and discrete state text<br>l. Red and yellow limit values | | |
| DM4 | The SUG shall include record formats for information about each of the following command elements:<br>a. CCSDS commands<br>b. Non-CCSDS commands<br>c. Command data area parameters<br>d. Command data area parameter conversions | | |
| DM5 | The SUG shall include record formats for the information required to map telemetry verifiers to commands received | | |

## 2.4 Data Transport Requirements

These requirements concern sending and receiving or re-formatting of input and output data.

| *DT* | *Data Transport Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| DT1 | The SIMSS shall provide a module that is capable of sending data via TCP/IP and UDP/IP. | Full | |
| DT1.1 | This module shall be capable of connecting to an IP socket for the purpose of transmitting data. | Full | |
| DT1.2 | This module shall be capable of transmitting data over an IP socket in UDP unicast mode. | Full | |
| DT1.3 | This module shall be capable of transmitting data over an IP socket in UDP multicast mode. | Full | |
| DT1.4 | This module shall be capable of transmitting data over an IP socket in TCP mode. | Full | |
| DT1.4.1 | This module shall be capable of transmitting data over an IP socket as a TCP/IP client. | Full | |
| DT1.4.2 | This module shall be capable of transmitting data over an IP socket as a TCP/IP server. | Full | |
| DT1.5 | This module shall be capable of transmitting up to 6000 bytes of data in a single IP data block. | Full | |

| | | | |
|---|---|---|---|
| DT1.6 | This module shall be capable of interfacing with other modules for the purpose of accepting data to be transmitted. | Full | |
| DT1.7 | This module shall be capable of displaying the following IP interface status information to the user:<br>a. Number of packets transmitted<br>b. Enabled/disabled status<br>c. The most recent data that was transmitted | Full | |
| DT1.8 | This module shall provide the user the capability of setting, by means of a user interface, IP socket parameters including the following:<br>a. IP address<br>b. Port number<br>c. Defined or variable data size<br>d. Multicast address | Full | |
| DT2 | The SIMSS shall provide a module that is capable of receiving data via TCP/IP and UDP/IP. | Full | |
| DT2.1 | This module shall be capable of connecting to an IP socket for the purpose of receiving data. | Full | |
| DT2.2 | This module shall be capable of receiving data over an IP socket in UDP mode. | Full | |
| DT2.3 | This module shall be capable of receiving data over an IP socket in UDP multicast mode. | Full | |
| DT2.4 | This module shall be capable of receiving data over an IP socket in TCP mode. | Full | |
| DT2.4.1 | This module shall be capable of receiving data over an IP socket as a TCP/IP client. | Full | |
| DT2.4.2 | This module shall be capable of receiving data over an IP socket as a TCP/IP server. | Full | |
| DT2.5 | This module shall be capable of receiving up to 6000 bytes of data in a single IP data block. | Full | |
| DT2.6 | This module shall be capable of interfacing with other modules for the purpose of passing on received data. | Full | |
| DT2.7 | This module shall be capable of displaying the following IP interface status information to the user:<br>a. Number of packets received<br>b. Enabled/disabled status<br>c. The most recent data that was received | Full | |
| DT2.8 | This module shall provide the user the capability of setting, by means of a user interface, IP socket parameters including the following:<br>a. IP address<br>b. Port number<br>c. Defined or variable data size<br>d. Multicast address | Full | |
| DT3 | The SIMSS shall provide a module that is capable of sending | Full | |

| | | | |
|---|---|---|---|
| | serial data and clock using ISA based architecture. | | |
| DT3.1 | This module shall be capable of connecting to a serial line for the purpose of transmitting data. | Full | |
| DT3.2 | This module shall be capable of transmitting a frame length up to 4096 bytes of data. | Full | |
| DT3.3 | This module shall be capable of interfacing with other modules for the purpose of accepting data to be transmitted. | Full | |
| DT3.4 | This module shall be capable of selecting the internal clock in a range from 100 Hz to 4 MHz. | Full | |
| DT3.5 | This module shall be capable of displaying the following information to the user<br>a. Enabled/disabled status<br>b. Data frequency<br>c. Frame length<br>d. Frame count<br>e. The most recently transmitted block of data | Full | |
| DT3.6 | This module shall provide the user the capability of setting serial interface board parameters through a user interface including the following encoding choices:<br>a. Non-Return-to-Zero-Level (NRZ-L) (true or inverted)<br>b. NRZ-Mark (NRZ-M) (true or inverted)<br>c. NRZ-Space (NRZ-S) (true or inverted)<br>d. Bi-phase-Level (BI0-L)<br>e. BI0-M<br>f. BI0-S | Full | |
| DT3.7 | This module shall be capable of providing an external clock interface in the RS422/TTL standard. | Full | |
| DT4 | The SIMSS shall provide a module that is capable of receiving serial data and clock (RS422/TTL), using ISA based architecture. | Full | |
| DT4.1 | This module shall be capable of connecting to a serial line for the purpose of receiving data. | Full | |
| DT4.2 | This module shall be capable of receiving a frame length of up to 4096 bytes of data. | Full | |
| DT4.3 | This module shall be capable of interfacing with other modules for the purpose of passing on received data. | Full | |
| DT4.4 | This module shall be capable of displaying the following information to the user:<br>a. Enabled/disabled status<br>b. Data frequency<br>c. Frame length<br>d. Frame count<br>e. Subframe count<br>f. Frame sync drop count | Full | |

| | | | |
|---|---|---|---|
| | g. Subframe drop count or sequence drop<br>h. Frame sync search and lock status<br>i. The most recently received block of data | | |
| DT4.5 | This module shall provide the user the capability of setting serial interface board parameters through a user interface including the following choices:<br>a. NRZ-L (true or inverted)<br>b. NRZ-M (true or inverted)<br>c. NRZ-S (true or inverted) | Full | |
| DT4.6 | This module shall provide the user the following information:<br>a. Sync status (Lock, Search, Idle)<br>b. Subframe status (Lock, Search, Idle)<br>c. Clock status (active or inactive)<br>d. Frame count<br>e. Frame drop count<br>f. Subframe count<br>g. Subframe drop count<br>h. Data orientation (most significant bit or least significant bit is transmitted first)<br>i. Data polarity (true or inverted)<br>j. Clock polarity (true or inverted)<br>k. Auto polarity check<br>l. Frame length<br>m. RS422/TTL<br>n. Correlation<br>o. Subframe size<br>p. Subframe location<br>q. Subframe start count<br>r. Subframe stop count<br>s. Sync size (up to 8 bytes)<br>t. Sync mask<br>u. Sync pattern | Full | |
| DT4.7 | This module shall provide the capability of operating in asynchronous mode | Full | |
| DT5 | The SIMSS shall provide a module that is capable of sending serial data and clock (RS422) on with PCI based serial cards. | Full | |
| DT5.1 | This module shall be capable of connecting to a serial line for the purpose of transmitting data. | Full | |
| DT5.2 | This module shall be capable of transmitting up to 4096 bytes of data in a single operation. | Full | |
| DT5.3 | This module shall be capable of interfacing with other modules for the purpose of accepting data to be transmitted. | Full | |
| DT5.4 | This module shall be capable of selecting the internal clock | Full | |

| | | | |
|---|---|---|---|
| | in a range from 900 Hz to 4 MHz. (Output internal clock only. Input clock is driven by the external clock). | | |
| DT5.5 | This module shall be capable of selecting the channel to transmit data. | Full | |
| DT5.6 | This module shall be capable of selecting the polarity of the data stream (true or inverted). | Full | |
| DT5.7 | This module shall be capable of selecting data orientation (most significant bit or least significant bit is transmitted first). | Full | |
| DT5.8 | This module shall provide the user the capability of setting serial interface board parameters through a user interface including the following Pulse Code Modulation (PCM) code choices:<br>a. NRZ-L (true or inverted)<br>b. NRZ-M (true or inverted)<br>c. NRZ-S (true or inverted)<br>d. BI0-L (true or inverted)<br>e. BI0-M (true or inverted)<br>f. BI0-S (true or inverted) | Full | |
| DT5.9 | This module shall be capable of selecting data encoding (CRC, Reed-Solomon, Randomization, Convolution) | Full | |
| DT5.10 | This module shall be capable of displaying the following information to the user<br>a. PCM code selection<br>b. Data Orientation<br>c. Channel<br>d. Clock Type<br>e. Clock frequency<br>f. Data polarity<br>g. Frame length<br>h. Frame count<br>i. Data encoding selection<br>j. The most recently transmitted block of data | Full | |
| DT5.11 | This module shall be capable of providing an external clock interface in the RS422 standard. | | |
| DT6 | The SIMSS shall provide a module that is capable of receiving serial data and clock (RS422) with PCI based serial cards. | Full | |
| DT6.1 | This module shall be capable of connecting to a serial line for the purpose of receiving data. | Full | |
| DT6.2 | This module shall be capable of receiving up to 4096 bytes of data in a single operation. | Full | |
| DT6.3 | This module shall be capable of interfacing with other modules for the purpose of passing on received data. | Full | |

| | | | |
|---|---|---|---|
| DT6.4 | This module shall be capable of providing an external clock interface in the RS422 standard. | Full | |
| DT6.5 | This module shall be capable of selecting data orientation (most significant bit or least significant bit is received first). | Full | |
| DT6.6 | This module shall be capable of selecting data polarity (true, inverted, or auto polarity check). | Full | |
| DT6.7 | This module shall be capable of selecting synchronized patterns up to 4 bytes. | Full | |
| DT6.8 | This module shall be capable of selecting FIFO size up to 99 buffers | Full | |
| DT6.9 | This module shall be capable of selecting data input type (PDP and SIM). | Full | |
| DT6.10 | This module shall be capable of selecting of setup command information (Tail sequence, tail length, tail pattern, command size, and max command size). | Full | |
| DT6.11 | This module shall be capable of passing to next module a commands block in multiple packets or one packet, specified by command size and max command size information. | Full | |
| DT6.12 | This module shall be capable of setting up maximum value, size in bits, and starting location in bits, of sub frame. | Full | |
| DT6.13 | This module shall provide the user the capability of setting serial interface board parameters through a user interface including the following choices:<br>a. NRZ-L (true or inverted)<br>b. NRZ-M (true or inverted)<br>c. NRZ-S (true or inverted) | Full | |
| DT6.14 | This module shall be capable of displaying the following information to the user:<br>a. Number of frames received<br>b. Number of frames dropped<br>c. Number of subframe received<br>d. Number of subframe dropped<br>e. PCM coding<br>f. Data polarity<br>g. Data orientation<br>h. Receiving channel<br>i. Data input type<br>j. Sync. Pattern<br>k. Sync. Size<br>l. Frame size<br>m. FIFO size<br>n. Tail length<br>o. Tail pattern<br>p. Tail sequence enabled/disabled status | Full | |

| | | | |
|---|---|---|---|
| | q. Maximum command size<br>r. Maximum value of subframe counter<br>s. Minimum value of subframe counter<br>t. Subframe counter size in bits<br>u. Subframe starting location in bits<br>v. Status of frame synchronization and subframe sequence<br>w. Data pattern within one frame<br>x. The most recently received block of data | | |
| DT7 | The SIMSS shall be able to operate in IP and serial modes simultaneously. | Full | |
| DT8 | The SIMSS shall support IP to Serial and Serial to IP data conversion and buffering. | Full | |
| DT9 | The SIMSS shall provide a module that is capable of using two Ethernet cards simultaneously. | Full | |
| DT9.1 | This module shall be able to select which Ethernet card is default card | Full | |
| DT9.2 | This module and the Ethernet card it is using shall be able to communicate independently of any other Ethernet card in the system. This includes requirements DT1.1 – DT1.8, DT2.1-DT2.8. | Full | |
| DT9.3 | This module shall be able to connect to any Ethernet, independent of any other card. | Full | |
| DT10 | The SIMSS shall provide a module that is capable of formatting data into message blocks with standard or user-defined formats. | Partial | |
| DT10.1 | This module shall be capable of connecting to other modules for the purpose of receiving data to be formatted. | Full | |
| DT10.2 | This module shall be capable of connecting to other modules for the purpose of passing along formatted data. | Full | |
| DT10.3 | This module shall be capable of creating messages with the following formats (this implies the capability to block data into these formats):<br>a. NASA Communications (NASCOM) Johnson Space Center (JSC) blocks<br>b. NASCOM Multiplexer-Demultiplexer (MDM) blocks<br>c. NASCOM Deep Space Network (DSN) block<br>d. NASCOM DSN/GSFC Interface Blocks (DGIB)<br>e. NASCOM JSC to Payload Operations Control Center (POCC) Blocks<br>f. NASCOM JSC to Ground Space Tracking Data Network (GSTDN) Blocks<br>g. Real-Time Protocol (RTP) messages<br>h. EOS Data Operations System (EDOS) ground | Partial | a, b, c, d, e, f, g only;<br>a includes POCC to JSC command blocks. |

| | | | |
|---|---|---|---|
| | message header messages<br>i. EDOS service header messages | | |
| DT10.4 | This module shall be capable of adding message headers and trailers based on the contents of a text file providing formatting information according to SIMSS specifications. | Full | Includes CCS T3ICD format |
| DT10.5 | This module shall provide a user interface that allows the user to set the following parameters:<br>a. The type of message to be formatted and configurable parameters associated with that type<br>b. The pathname of the file to be used as the basis of formatting | Partial | b only |
| DT10.6 | This module shall provide a status display that shows the user the following information:<br>a. The number of data packets received<br>b. The number of data blocks sent<br>c. The current type of message being formatted<br>d. Configuration parameters<br>e. Last block transmitted | Partial | a, b only |
| DT10.7 | This module shall provide the capability to transmit Circuit Assurance Blocks (CABs). The CAB blocks will be transmitted at a rate of 1 block per five seconds when no telemetry blocks are active. | | |
| DT10.8 | This module shall provide the capability to transmit 'empty' NASCOM blocks in the configured block type. The 'empty' blocks shall be defined as NASCOM blocks with a data length and rate consistent with the configured telemetry rate where the data content is fill data. | | |
| DT11 | The SIMSS shall provide a module that is capable of validating message blocks with standard or user-defined formats and extracting and sending the enclosed data on to another SIMSS module. | Partial | |
| DT11.1 | This module shall be capable of connecting to other modules for the purpose of receiving message data to be validated. | Full | |
| DT11.2 | This module shall be capable of connecting to other modules for the purpose of sending message data that has been extracted from data received. | Full | |
| DT11.3 | This module shall be capable of validating messages with the following formats:<br>a. NASA Communications (NASCOM) Johnson Space Center (JSC) blocks<br>b. NASCOM Multiplexer-Demultiplexer (MDM) blocks<br>c. NASCOM Deep Space Network (DSN) blocks<br>d. NASCOM DSN/GSFC Interface Blocks (DGIB)<br>e. NASCOM JSC to Payload Operations Control | Partial | a, b, c, d, e, f, g only;<br>a includes POCC to JSC command blocks. |

| | | | |
|---|---|---|---|
| | Center (POCC) Blocks<br>f. NASCOM JSC to Ground Space Tracking Data Network (GSTDN) Blocks<br>g. Real-Time Protocol (RTP) messages<br>h. EOS Data Operations System (EDOS) ground message header messages<br>i. EDOS service header messages | | |
| DT11.4 | This module shall be capable of validating message headers and trailers based on the contents of a text file providing formatting information according to SIMSS specifications. | Full | Includes CCS T3ICD format |
| DT11.5 | This module shall provide a user interface that allows the user to set the following parameters:<br>a. The type of message to be formatted and configurable parameters associated with that type<br>b. The pathname of the file to be used as the basis of formatting | Partial | Pathname only |
| DT11.6 | This module shall provide a status display that shows the user:<br>a. The number of data blocks received<br>b. The number of data packets sent<br>c. The current type of message being validated<br>d. Configuration parameters<br>e. Last block received | Partial | a, b only |
| DT11.7 | This module shall provide the capability to monitor the reception of CABs on selected input channels. | | |
| DT11.8 | This module shall provide the capability to monitor the reception of 'empty' blocks on selected input channels. | | |
| DT12 | The SIMSS shall provide a module that is capable of encoding serial data via software and of transmitting the resulting encoded data. | Full | |
| DT12.1 | This module shall be capable of adding 16-bit cyclic redundancy check (CRC-16) encoding to a serial data stream. | Full | |
| DT12.2 | This module shall be capable of adding Reed-Solomon check symbols to a serial data stream. | Full | |
| DT12.3 | This module shall be capable of adding convolutional encoding to a serial data stream. | Full | |
| DT12.4 | This module shall be capable of adding randomization encoding to a serial data stream. | Full | |
| DT12.5 | This module shall allow the user to specify the transmission encoding methods to be performed on a serial data stream. | Full | |
| DT12.6 | This module shall allow the user to specify the data block size to be encoded. | Full | |
| DT12.7 | This module shall be capable of displaying the status and configuration information to the user, including: | Full | |

| | | | |
|---|---|---|---|
| | a. The current encoding methods being performed on the data stream<br>b. The contents of the last data block encoded<br>c. The total number of data blocks transmitted | | |
| DT13 | The SIMSS shall provide a module that is capable of supporting command echo. | Partial | |
| DT13.1 | This module shall be capable of connecting to another module for the purpose of receiving data to be echoed. | Full | |
| DT13.2 | This module shall be capable of connecting to another module for the purpose of sending data that has been echoed. | Full | |
| DT13.3 | This module shall be capable of extracting the source and destination identifiers from a data block received, swapping them, re-generating the CRC for the block if applicable, and sending out the resulting block. | Full | |
| DT13.4 | This module shall provide the user with the capability to set the following parameters:<br>a. Location of the source identifier in the block<br>b. Size of the source identifier<br>c. Location of the destination identifier in the block<br>d. Size of the destination identifier | Full | Settable from a configura-tion file. |
| DT13.5 | This module shall provide the following status information to the user:<br>a. Number of blocks received<br>b. Current location being used for the source identifier<br>c. Current location being used for the destination identifier<br>d. Last source and destination identifiers received<br>e. Contents of the most recent block received | Partial | a and e only. It is questionable whether the other items are needed. |
| DT14 | The SIMSS shall support continuous, intermittent, and discrete transmission modes. | | |

## 2.5 Database Flat File Utility Requirements

| *DU* | *Database Flat File Utility Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| DU1 | There shall be stand alone utilities for converting project specific database flatfiles into a binary file to be read in by the Generic CCSDS telemetry generating module. | Full | |
| DU2 | The binary file generated by the SIMSS utilities will be syntactically valid. | Partial | Not all validations implemented |
| DU3 | There shall be a SIMSS utility to convert project specific database flatfiles into an ASCII text file. | Full | |
| DU3.1 | The ASCII utility shall be able to read in multiple files. These | Full | |

| | | | |
|---|---|---|---|
| | files shall include flatfiles from a specific project. | | |
| DU3.2 | The user shall be able to define the following for each input file:<br>a. Column delimiter.<br>b. Number of fields per record.<br>c. The relationship between the input file fields and the output file. | Full | |
| DU3.3 | The user shall be able to define the following (global to all telemetry):<br>a. If packets are allowed to cross VCDUs or transfer frames.<br>b. If transmission of VCDU will wait for VCDU to be filled.<br>c. SCID.<br>d. Size of VCDU or transfer frame. | Full | |
| DU4 | There shall be a SIMSS utility to convert an ASCII text file into a binary file. | Full | |
| DU4.1 | The ASCII-to-binary utility will take a single ASCII text as input and output binary file(s) as needed by the Generic CCSDS telemetry generating module. | Full | |
| DU4.2 | The file will define the following (global to all telemetry):<br>a. If packets are allowed to cross VCDUs or transfer frames.<br>b. If transmission of VCDU will wait for VCDU to be filled.<br>c. Version (VCDU or transfer frame).<br>d. SCID.<br>e. Size of VCDU or transfer frame. | Partial | Transfer Frames not implemented |
| DU4.3 | The file will define the following (for VCDUs):<br>a. VCID<br>b. Output Channel<br>c. Replay Flag<br>d. Error Control Flag (in primary header)<br>e. Insert Zone Flag<br>f. Insert Zone Size<br>g. Op Control Flag<br>h. Error Control Flag (not in primary header)<br>i. RS Encoding Flag<br>j. RS Encoding Size<br>k. Command Channel | Full | |
| DU4.4 | The file will define the following (for transfer frames):<br>a. VCID<br>b. Master Channel<br>c. Secondary Header Flag<br>d. Secondary Header Length<br>e. Op Control Flag<br>f. Frame Error Control Flag<br>g. Output Channel<br>h. Sync Flag<br>i. Packet Order Flag<br>j. Segment Length ID<br>k. Command Channel | None | Transfer Frames not implemented |
| DU4.5 | The file will define the following (for packets):<br>a. APID<br>b. VCID<br>c. Secondary Header Type<br>d. Size<br>e. Interval | Full | |

| | | | |
|---|---|---|---|
| | f. Time Offset<br>g. Skey<br>h. Key Offset<br>i. Key Length | | |
| DU4.6 | The file will define the following (for mnemonics)<br>a. Name<br>b. Type<br>c. Description<br>d. APID<br>e. Skey<br>f. Bit Offset<br>g. Bit Size | Full | ParmID was removed |
| DU4.7 | In case where the project information is incomplete, the utility shall provide default values for all non-mandatory unpopulated fields. | Full | |

## 2.6 Telemetry Generation Requirements

Requirements regarding telemetry generation, transmission, and display.

| *TG* | *Telemetry Generation Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| TG1 | The SIMSS shall provide a module that generates CCSDS telemetry based on information stored in a standard database. | Full | |
| TG1.1 | This module shall generate telemetry packets according to CCSDS standards. | Partial | AOS only |
| TG1.1.1 | This module shall provide a standard packet primary header with an incrementing packet counter. | Full | |
| TG1.1.2 | This module shall provide the database-driven option of a secondary header in the one of the following formats:<br>a. SMEX secondary header<br>b. EOS secondary header | | |
| TG1.1.3 | This module shall use the database to define the following packet-level parameters:<br>a. Application id<br>b. Source<br>c. Existence of a secondary header<br>d. Type of the secondary header<br>e. Length of packet<br>f. Virtual Channel id<br>g. Interval<br>h. Interval offset<br>i. Key<br>j. Key offset<br>k. Key length | Partial | b, d not implemented |
| TG1.1.4 | This module shall provide the user with the capability to:<br>a. Set any value in a packet primary header<br>b. Set any value in a packet secondary header | Partial | All except b. |

| | | | |
|---|---|---|---|
| | c. Set any value in the data area of a packet<br>d. Set a pattern in the data area of a packet<br>e. Enable or disable the sending of a packet on a timed basis<br>f. Change the interval between sending out packets | | |
| TG1.1.5 | This module shall be capable of displaying to the user:<br>a. A packet's primary header<br>b. A packet's secondary header<br>c. A packet's data area<br>d. The interval at which a packet is being sent out | Partial | All except b, d |
| TG1.2 | This module shall be capable of packing packets into transfer frames or VCDUs and sending them on a virtual channel according to CCSDS or CCSDS AOS standards. | Partial | AOS only |
| TG1.2.1 | This module shall be capable of either splitting or not splitting packets across frames based on parameters in the database. | Full | |
| TG1.2.2 | This module shall generate a frame header with an incrementing frame count. | Full | |
| TG1.2.3 | This module shall use the database to define the following transfer frame and virtual channel parameters:<br>a. Virtual channel identifier<br>b. Frame size<br>c. Packet splitting flag<br>d. Standard or AOS flag<br>e. Mapping of packets to virtual channels<br>f. Cross packets<br>g. Whole VCDU or partial<br>h. Spacecraft id<br>i. Size (excluding RS)<br>j. Output channel<br>k. Replay flag<br>l. Header error control flag<br>m. Insert zone size<br>n. Op control flag<br>o. Error control flag<br>p. RS encoding size<br>q. Command channel | Partial | AOS only. d,l,m,n not implement ed. |
| TG1.2.4 | This module shall use the database to define the following telemetry point parameters:<br>a. Mnemonic<br>b. length<br>c. parameter id<br>d. type<br>e. description | Full | |
| TG1.2.5 | This module shall use the database to define the following telemetry location parameters: | Full | |

| | | | |
|---|---|---|---|
| | a. apid<br>b. key<br>c. parameter id<br>d. bit offset<br>e. bit size<br>f. mnemonic | | |
| TG1.2.6 | This module shall provide the user with the capability to:<br>a. Set any value in the frame header for a virtual channel, either once or constantly until disabled<br>b. Set any value in the data area of a frame for a virtual channel, either once or constantly until disabled<br>c. Change the mapping of packets to virtual channels<br>d. Enable or disable a virtual channel | Partial | a, b are implemented |
| TG1.2.7 | This module shall be capable of displaying to the user:<br>a. The most recent frame header for a virtual channel<br>b. The most recent frame contents for a virtual channel<br>c. The packet mapping for a virtual channel<br>d. The number of packets received for a virtual channel<br>e. The number of frames sent on a virtual channel | | |
| TG1.3 | This module shall be capable of combining the frames for a virtual channel into a physical channel according to CCSDS standards. | Full | |
| TG1.3.1 | This module shall be capable of sending telemetry data over one to three physical channels, each of which shall correspond to a SIMSS channel or link. | Full | |
| TG1.3.2 | This module shall use the database to define the following physical channel parameters:<br>a. Number of physical channels<br>b. Virtual channel to physical channel mapping<br>c. Whether the frame for the virtual channel must be full to be transmitted | Full | |
| TG1.3.3 | This module shall provide the user with the capability to:<br>a. Enable or disable a physical channel<br>b. Change the virtual channel to physical channel mapping<br>c. Change the must-be-full flag for a virtual channel | Partial | a only, c only available in the flat file input data configura-tion stage |
| TG1.3.4 | This module shall be capable of displaying to the user:<br>a. The current number of physical channels<br>b. Whether each channel is enabled or disabled<br>c. The virtual channels mapped to a physical channel<br>d. The priority of a virtual channel on a physical channel<br>e. The must-be-full flag for a virtual channel<br>f. The number of frames sent over a physical channel | Partial | a, b and f only |

| | | | |
|---|---|---|---|
| | g. The most recent frame sent over a physical channel | | |
| TG1.4 | This module shall be capable of generating fill packets and frames as necessary according to CCSDS standards and database parameters. Virtual channel 7 (for standard CCSDS) and virtual channel 63 (for CCSDS AOS) shall be channels consisting entirely of fill frames. | Partial | AOS only |
| TG1.5 | This module shall be capable of handshaking with the serial or another external module for the purpose of sending out data at a rate defined at that module. | Full | |
| TG1.6 | This module shall allow the ability to select a database. | Full | |
| TG2 | The SIMSS shall provide a module that generates time-division multiplexed (TDM) telemetry based on information by the user. | Partial | |
| TG2.1 | This module shall generate multiple telemetry formats on a minor frame/major frame basis. | Full | One format at a time |
| TG2.2 | This module shall be capable of sending telemetry over 1 channel. | Full | |
| TG2.3 | This module shall use a configuration file or GUI to define the following parameters:<br>a. Number of minor frames per major frame<br>b. Size of a minor frame<br>c. Position of the minor frame counter in the minor frame<br>d. Position of the major frame counter in the minor or major frame | Full | a-d settable by user. |
| TG2.4 | This module shall be capable of adding valid CRC check words to the end of a minor frame. | Full | 16 or 32 bits |
| TG2.5 | This module shall be capable of accepting the value of any parameter (e.g., a command counter) from an external module. | Full | Currently on byte boundary |
| TG2.6 | This module shall provide the user with the capability to:<br>a. Enable or disable the physical channel<br>b. Set any value in a minor frame<br>c. Set patterns in a minor frame or sequence of minor frames, either consecutive or subcommutated<br>d. Enable or disable setting the CRC check words | Partial | a-d (no subcoms) |
| TG2.7 | This module shall be capable of displaying to the user:<br>a. Whether each channel is enabled or disabled<br>b. The contents of the most recent minor frame to be sent from that physical channel<br>c. The minor and major frame counts for a physical channel<br>d. The total number of frames sent out over the physical | Partial | a, c, d only. b use Monitor or Test-Module to view. |

| | | | |
|---|---|---|---|
| | channel | | |
| TG2.8 | This module shall be capable of handshaking with the serial or another external module for the purpose of sending out data at a rate defined at that module. | Full | |
| TG3 | The SIMSS shall provide a generic, data-driven module that is capable of modifying TDM telemetry. | Full | |
| TG3.1 | This module shall be capable of ingesting a data stream containing only TDM minor frames, perform modifications on the data, and output the modified data. | Full | |
| TG3.2 | This module shall provide the user with the capability to define the:<br>a. Minor frame size in bytes.<br>b. Minor frame counter size in bits.<br>c. Minor frame counter location at the bit level<br>d. If minor frame counter is bit flipped. | Full | |
| TG3.3 | This module shall be capable of providing mod-bit functionality, up to 32 consecutive bits. | Full | |
| TG3.3.1 | This module shall be capable of displaying to the user:<br>a. The current minor frame number.<br>b. The data value in the telemetry stream before modification.<br>c. The data value in the telemetry stream after modification. | Full | |
| TG3.3.2 | This module shall provide the user with the capability to:<br>a. Enable or disable any modification.<br>b. Define the start frame.<br>c. Define the subcom depth.<br>d. Define the start byte.<br>e. Define the start bit.<br>f. Define the bit length.<br>g. Define the value to be placed in the minor frame. | Full | |
| TG3.4 | This module shall be capable of modifying telemetry via the use of defined telemetry mnemonics. | Full | |
| TG3.4.1 | This module shall be capable of receiving and implementing telemetry mnemonic updates provided by a modeling interface module. | Partial | |
| TG3.4.2 | This module shall be capable of ingesting an ASCII text file that contains a list of telemetry mnemonics and their telemetry locations. | Full | |
| TG3.4.3 | This module shall provide the user the capability to modify all aspects of the database during runtime. | Full | |
| TG3.4.4 | This module shall provide the user the capability to set individual telemetry mnemonics to PCM (or raw) values. | Full | |
| TG3.5 | This module shall be capable of receiving and implementing telemetry mnemonic updates provided by a command ingest | Full | |

| | | | |
|---|---|---|---|
| | module. | | |
| TG3.6 | This module shall be fully controllable from the scenario module. | Full | |

## 2.7 Command Ingest Requirements

Requirements regarding command ingest, response, and display.

| *CI* | *Command Ingest Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| CI1 | The SIMSS shall provide a module with the capability to receive, validate, and identify CCSDS commands. | Partial | |
| CI1.1 | This module shall be capable of receiving commands in the form of CLTUs. | Full | |
| CI1.2 | This module shall be capable of performing the following CCSDS validation checks:<br>a. CLTU header and trailer<br>b. Codeblock CRC<br>c. Transfer frame header<br>d. Frame Acceptance and Reporting Mechanism<br>e. Packet primary header | Full | |
| CI1.3 | This module shall generate a command link control word (CLCW) for each virtual channel that reflects the commands received. | Full | |
| CI1.4 | This module shall be capable of receiving and executing CCSDS FARM special commands, including:<br>a. Unlock FARM<br>b. Set expected frame sequence number | Full | |
| CI1.5 | This module shall be capable of using a database to determine if a command received is valid. | | |
| CI1.6 | This module shall provide the user with the capability to enable or disable any validation check. | Full | |
| CI1.7 | This module shall provide the user with the capability to set or change the following parameters:<br>a. Expected CLTU header<br>b. Expected CLTU trailer<br>c. Codeblock size<br>d. Spacecraft id (SCID)<br>e. Any field in the CLCWs, including<br>1. Virtual channel id (VCID)<br>2. Next expected frame sequence number<br>f. FARM sliding window size<br>g. Database source and version to use for validation and identification | Partial | g. not implemented |
| CI1.8 | This module shall be capable of displaying to the user: | Full | |

| | | | |
|---|---|---|---|
| | a. The most recent CLTU received<br>b. The codeblock data areas from the most recent CLTU received<br>c. The most recent transfer frame received for each virtual channel<br>d. The most recent packet received for each virtual channel<br>e. The current CLCW for each virtual channel<br>f. Counts of valid and invalid command elements received | | |
| CI1.9 | This module shall be capable of generating event messages when errors are seen that provide specifics on the error. | Full | |
| CI1.10 | This module shall be capable of generating an event message when a valid command is received that provides the command mnemonic and information on its contents. | | |
| CI1.11 | This module shall provide the user with the capability to suspend/resume the update of CLCW upon completion of command validation. | Full | |
| CI1.12 | This module shall be capable of providing the current CLCW for any virtual channel to another module. | Full | |
| CI2 | The SIMSS shall provide a module with the capability to receive, validate, and identify non-CCSDS commands. | Full | |
| CI2.1 | This module shall be capable of receiving a stream of non-CCSDS commands. | Full | |
| CI2.2 | This module shall be capable of performing the following validation checks on non-CCSDS commands:<br>a. Spacecraft identifier<br>b. Hamming code<br>c. Command block format including barker code, preamble, and postamble | Full | |
| CI2.3 | This module shall be capable of ingesting a database file containing commands and associated telemetry verifiers. These verifiers and their values shall be forwarded to another module for telemetry updates. | Full | |
| CI2.4 | This module shall provide the user with the capability to enable or disable any validation check. | Full | |
| CI2.5 | This module shall provide the user with the capability to set or change the following parameters:<br>a. Expected spacecraft identifier<br>b. Expected barker code<br>c. Expected pre-amble and post-amble pattern<br>d. Expected pre-amble and post-amble length<br>e. Internal command counter<br>g. Location(s) of command counter in telemetry | Full | Option f deleted. |

| | | | |
|---|---|---|---|
| CI2.6 | This module shall be capable of displaying to the user:<br>a. The most recent command received<br>b. Counts of valid and invalid command elements received | Full | |
| CI2.7 | This module shall be capable of generating event messages when errors are seen that provide specifics on the error. | Full | |
| CI2.8 | This module shall be capable of generating an event message when a valid command is received that provides the command mnemonic and information on its contents. | Full | Req. needs rewording |
| CI2.9 | This module shall provide the user with the capability to suspend/resume the update of command counter(s) upon completion of command validation. | Full | |
| CI2.10 | This module shall be capable of providing the current command counter(s) to another module. | Full | |

## 2.8 Command Generation Requirements

Requirements regarding command generation, transmission, and display.

| *CG* | *Command Generation Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| CG1 | The SIMSS shall provide a module that is capable of creating, saving, reading, modifying, and transmitting telecommand headers and binary files under user control. | Partial | |
| CG1.1 | The module shall support the following predefined types of CCSDS data buffers:<br>a. Transfer frame header<br>b. Packet primary header<br>c. Packet secondary header<br>d. Command data<br>e. Command | Partial | |
| CG1.1.1 | The module shall allow the user to construct a data buffer of a predefined type (CG1.1a-e) and save it as a binary data file. | Partial | |
| CG1.1.2 | The module shall allow the user to identify a binary data file as a predefined type (CG1.1a-e) and will interpret the file accordingly. | | |
| CG1.1.3 | The module shall allow the user to identify a portion of a binary data file by the start byte and number of bytes as a predefined type (CG1.1a-e) and will interpret the portion of the file accordingly. | Partial | |
| CG1.1.4 | The module shall allow the user to change any field in a predefined type (CG1.1a-e) buffer. | Full | |
| CG1.1.5 | The module shall automatically increment counter fields in a predefined type (CG1.1a-e) buffer unless overridden by the user. | Full | |
| CG1.2 | The module shall be capable of generating CCSDS composite files. | Full | |
| CG1.2.1 | The module shall allow the user to combine separate files (from the predefined list of data type CG1.1a-e) into a single file. (For example, a command may consist of a transfer frame header, a packet primary header, a packet secondary header, and command data.) | Full | |
| CG1.2.2 | The module shall be capable of displaying to the user the individual components of a composite file and the identity information for each of those components. | | |
| CG1.2.3 | The module shall maintain identify information about any binary data file that contains other than raw, noncomposite data. This identify information shall be kept separate from the data file and shall indicate what the file represents if other | Full | |

| | | | |
|---|---|---|---|
| | than raw data. | | |
| CG1.3 | The module shall be capable of processing raw data files containing CCSDS-formatted command data. | | |
| CG1.3.1 | The module shall be capable of converting a raw data file into codeblocks according to CCSDS specifications. | | |
| CG1.3.2 | The module shall be capable of calculating CRC and any polynomial check defined in the CCSDS specification for a raw data file. | | |
| CG1.3.3 | The module shall be capable of calculating CRC and any polynomial check defined in the CCSDS specification for data entered by the user. | | |
| CG1.4 | The module shall be capable of processing raw data files containing non-CCSDS formatted command data. | | Implement-ed in TDM Cmd Gen Module |
| CG1.4.1 | The module shall be capable of adding a barker code and a hamming code to command data from a raw data file. | | Implement-ed in TDM CmdGen Module |
| CG1.4.2 | The module shall be capable of converting command data from a raw data file between NRZ-L and NRZ-M data formats. | | Implement-ed in TDM CmdGen Module |
| CG1.5 | The module shall be capable of transmitting CCSDS and non-CCSDS commands in real-time. | Partial | |
| CG1.5.1 | The module shall be capable of sending the contents of a file as selected by the user. | | |
| CG1.5.2 | The module shall provide the user with the capability to send data once. | | |
| CG1.5.2 | The module shall provide the user with the capability to send data for a fixed number of times at a user-defined interval. | | |
| CG1.5.3 | The module shall provide the user with the capability to send data at a user-defined interval until manually stopped. | | |
| CG1.5.4 | The module shall provide the same capabilities available for binary file generation and manipulation to an internal data buffer. | | |
| CG1.6 | The module shall be capable of generating and saving spacecraft/user profiles. | | |
| CG1.6.1 | The module shall be capable of generating a spacecraft/user profile that will contain spacecraft-specific CCSDS information including the following:<br>a. Codeblock size<br>b. Spacecraft id | | |
| CG1.6.2 | The module shall be capable of generating a spacecraft/user profile that will contain spacecraft-specific non-CCSDS | | Implement-ed in TDM |

| | | | |
|---|---|---|---|
| | information including the following:<br>a. Command length<br>b. Spacecraft id<br>c. Barker code<br>d. Location of hamming code<br>e. Input code (NRZ-L, NRZ-M)<br>f. Preamble and postamble | | CmdGen Module. |
| CG1.7 | The module shall provide the user with the capability of typing in a command mnemonic and submnemonics (if appropriate) and constructing a command therefrom by interfacing with the SIMSS commanding operations database (ODB). | | |
| CG2 | The SIMSS shall provide a module that is capable of creating, saving, reading, modifying, and transmitting TDM formatted commands and TDM command binary files under user control. | Full | TDMCmdGen |
| CG2.1 | The module shall be capable of adding a barker code, preamble, postamble and a hamming code to command data from a configuration file. | Full | |
| CG2.2 | The module shall be capable of adding a barker code, preamble, postamble and a hamming code to command data in a command buffer. | Full | |
| CG2.3 | The module shall be capable of generating a command buffer using the following parameters:<br>a. Preamble and postamble content<br>b. Preamble and postamble length<br>c. Barker code and secondary barker code<br>d. Inclusion of hamming encoding<br>e. List of commands, command contents, and command lengths | Full | |
| CG2.4 | The module shall be capable of converting command data between NRZ-L and NRZ-M data formats. | N/A | This belongs to the Encoding module. |
| CG2.5 | The module shall be capable loading and saving a sequence profile that will contain spacecraft-specific TDM information including the following:<br>a. Preamble and postamble content<br>b. Preamble and postamble length<br>c. Barker code and secondary barker code<br>d. Inclusion of hamming encoding<br>e. List of commands, command contents, and command lengths | Full | |
| CG3 | The module shall be capable of sending the contents of a command data file as selected by the user. | Full | Common to both CCSDS |

| | | | |
|---|---|---|---|
| | | | and TDM cmd generation |
| CG3.1 | The module shall provide the user with the capability to send data once. | Full | |
| CG3.2 | The module shall provide the user with the capability to send data for a fixed number of times at a user-defined interval. | Full | |
| CG3.3 | The module shall provide the user with the capability to send data at a user-defined interval until manually stopped. | Full | |
| CG3.4 | The module shall provide the same capabilities available for binary file generation and manipulation to an internal data buffer. | Full | |
| CG3.5 | The module shall provide the user with the capability of typing in a command mnemonic and submnemonics (if appropriate) and constructing a command therefrom by interfacing with the SIMSS commanding operations database (ODB). | N/A | This is CCSDS specific, and should be in the CG1 section. |

## 2.9 Data Analysis Requirements

Requirements regarding the analysis and display of "generic" input data.

| *DA* | *Data Analysis Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| DA1 | The SIMSS shall provide a generic, data-driven module that is capable of CCSDS telemetry VC quality monitoring and decommutation. (VC Processor) | Full | |
| DA1.1 | This module shall be capable of monitoring a telemetry stream in CCSDS or CCSDS-AOS format. | Full | CCSDS-AOS only |
| DA1.2 | This module shall be capable of transmitting user selected VCs. | Full | |
| DA1.3 | This module shall be capable of validating transfer frames or VCDUs, including CRC check, RS check, and (De) Randomization.<br>a. CRC error count<br>b. RS error count<br>c. RS uncorrectable error count<br>d. Randomization (not displayed)<br>e. VC sequence error | Full | VCDUs only |
| DA1.4 | This module shall be capable of validating the following fields in a transfer frame or VCDU header:<br>a. Version<br>b. Spacecraft ID<br>c. Virtual Channel ID<br>d. VCDU counter<br>e. Replay flag<br>f. Spare bits | | Validation of these parameters is not implemented in R5.0. |
| DA1.5 | This module shall be capable of displaying to the user:<br>a. The most recent transfer frame or VCDU received<br>b. The parsed header of the most recent transfer frame or VCDU received<br>c. The number of transfer frames or VCDUs received with and without errors<br>d. VCDU sequence error count<br>e. The number of transfer frames or VCDUs received for each virtual channel.<br>f. CRC error count<br>g. RS error count<br>h. RS uncorrectable error count | Partial | VCDU display only.<br>h. is not included in R5.0. |
| DA2 | a. This module shall be capable of extracting packets from the VCDU and display packet information on status screen. (Packet Processor) | Full | |
| DA2.1 | This module shall be capable of displaying to the user:<br>a. The most recent packet received with a given APID | Full | Packet displays will not be in |

| | | | |
|---|---|---|---|
| | b. The parsed header of the most recent packet received with a given APID<br>c. The number of packets received with a specific APID with and without errors<br>d. Time interval between specific APIDs | | this module, rather the Monitor module will be used to display the selected APID packets. |
| DA2.2 | This module shall provide the user with the capability to:<br>a. Define whether to expect packets<br>b. Define the valid packet APIDs<br>c. Define expected content values in specific packets by APID | Partial | Packet control only. |
| DA2.3 | This module shall be capable of using the database for the following information:<br>a. Whether the packets in a frame should be split between frames<br>b. Valid packet APIDs<br>c. Valid packet lengths<br>d. Valid packet header values<br>e. The locations of telemetry parameters within a packet | Partial | The function of a-e are implemented but not using database. |
| DA2.4 | This module shall have option to select which APIDs can be forwarded to the next module. | | Packets filtered via APID; but can not be forwarded at this time. |
| DA3 | The SIMSS shall provide a generic, data-driven module that is capable of TDM telemetry data quality monitoring. | Partial | |
| DA3.1 | This module shall be of capable of validating the following fields in a TDM telemetry stream:<br>a. Sync pattern<br>b. Minor frame count<br>c. Major frame count<br>d. User-defined parameters | Full | |
| DA3.2 | This module shall be capable of displaying to the user the following rate-dependent items:<br>a. The most recent minor frame received<br>b. The most recent minor frame counter seen<br>c. The most recent major frame counter seen<br>d. The number of minor frames received, with and without errors<br>e. The number of major frames received, with and without errors | Partial | e is not implemented. |

| | | | |
|---|---|---|---|
| | f. Telemetry data, in raw format, extracted from the stream based on user information about size and position<br>g. The current value of the command counter(s) | | |
| DA3.3 | This module shall provide the user with the capability to:<br>a. Enable or disable any element of the validation process<br>b. Define the size of a minor frame<br>c. Define the number of minor frames in a major frame using minimum and maximum value<br>d. Define the size, value, and position of the expected sync pattern<br>e. Define the size and position of the minor frame counter<br>f. Define the size and position of the major frame counter<br>g. Define the size, position, and subcommutation of parameters to display or validate<br>h. Save and restore the user-defined configuration | Full | |
| DA3.4 | This module shall be capable of extracting and forwarding data to another module, via no less than 3 output channels, defined in the following manner:<br>a. Minor frames – all, one individual frame, or using subcom depth<br>b. User-defined areas within minor frames by start byte and number of bytes | Full | |
| DA3.5 | This module shall provide the capability to ingest asynchronous normal and inverted data and sync align data using a maximum sync pattern of 32 bits. | | |
| DA4 | The SIMSS shall provide a module with a capability of displaying a data stream in operator selectable display formats. | Full | |
| DA4.1 | This module shall be capable of displaying data in decimal. | Full | |
| DA4.2 | This module shall be capable of displaying data in 8, 16, or 32 bit hexadecimal. | Full | |
| DA4.3 | This module shall be capable of displaying data in 8, 16, or 32 bit octal. | Full | |
| DA4.4 | This module shall be capable of shifting the displayed data 1-31 bits, left or right. | Full | |
| DA4.5 | This module shall be capable of inverting the displayed data. | Full | |
| DA4.6 | This module shall be capable of converting the displayed data to/from NRZ-L and NRZ-M formats. | Full | |
| DA5 | The SIMSS shall provide a module with a capability of encoding data from a data stream. | Partial | |
| DA5.1 | This module shall be capable of adding CRC encoding to a data stream. | Full | |
| DA5.2 | This module shall be capable of adding Reed-Solomon check | Full | |

| | | | |
|---|---|---|---|
| | symbols to a data stream. | | |
| DA5.3 | This module shall be capable of adding convolutional encoding to a data stream. | Full | |
| DA5.4 | This module shall be capable of adding randomization encoding to a data stream. | Full | |
| DA5.5 | This module shall be capable of de-randomizing data from a data stream. | Partial | Randomization is also de-randomization. Instead of creating additional decoder module, this shall be used as de-randomizer as well. |
| DA5.6 | This module shall provide the user with the capability to specify the encoding to be performed on a data stream. | Full | |
| DA5.7 | This module shall be capable of displaying status and configuration information to the user, including:<br>a. Frame Count<br>b. CRC-16 enabled/disabled<br>c. Convolution enabled/disabled<br>d. Randomization enabled/disabled<br>e. Reed-Solomon enabled/disabled<br>f. Interleave<br>g. Virtual Fill<br>h. Frame Size in bytes<br>i. Sync Pattern Size in bytes | Full | |

## 2.10 Timing Requirements

Requirements regarding time code(s) and interrupt generation.

| *TM* | *Timing Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| TM1 | The SIMSS shall be capable of using a timing card to drive NASA-36 time format | Full | Internal and external reference sync time |
| TM2 | The SIMSS shall be capable of generating time formats. | Partial | PB4 time format only |
| TM2.1 | The SIMSS shall be capable of generating GMT. | Full | |
| TM2.2 | The SIMSS shall be capable of generating UTC. | | |
| TM2.3 | The SIMSS shall be capable of generating UT1. | | |

| | | | |
|---|---|---|---|
| TM3 | The SIMSS shall be capable of formatting time data from one of the standard time formats into any of the following NASA standard time formats:<br>a. PB4<br>b. PB5<br>c. Small Explorer (SMEX) packet header time<br>d. CCSDS unsegmented time code | Partial | a only |
| TM4 | The SIMSS shall be capable of generating mission specific time formats created by library function. | Partial | Generic time format only – PB4 |
| TM5 | The SIMSS shall control timed activities in a simulated accelerated mode. | | |

## 2.11 Data Archiving Requirements

Requirements regarding storage (logging) and retrieval of input and output data.

| *DR* | *Data Archiving Requirements* | *.0 Impl.* | *Comments* |
|---|---|---|---|
| DR1 | The SIMSS shall provide a module capable of storing the contents of a data stream to disk files (i.e. logging). | Full | |
| DR1.1 | This module shall be capable of opening a disk file upon request from other modules. | Full | |
| DR1.2 | This module shall be capable of closing a disk file upon request from other modules. | Full | |
| DR1.3 | This module shall be capable of appending 6000 bytes of data to an open disk file in one operation. | Full | |
| DR1.4 | This module shall be capable of interfacing with other modules for the purpose of transferring data from those modules. | Full | |
| DR1.5 | This module will provide a user interface for the purpose of reporting status information including the following:<br>a. Enabled or disabled status of logging<br>b. Number of bytes written to a log file | Full | |
| DR1.6 | This module will provide a user interface for the purpose of entering the following information:<br>a. The maximum size of a log file<br>b. The name of a disk file | Full | |
| DR2 | The SIMSS shall provide a module that is capable of reading the contents of a disk file and sending it out as a data stream. | Partial | |
| DR2.1 | This module shall be capable of opening a disk file upon request from another module. | | |
| DR2.2 | This module shall be capable of closing a disk file upon request from another module. | | |
| DR2.3 | This module shall be capable of reading 6000 bytes of data from an open disk file in one operation. | Full | |
| DR2.4 | This module shall provide the user with the capability to set the following parameters:<br>a. Pathname of the file to read from the disk<br>b. Size (in bytes) of a block of data to read from the disk and send out at one time<br>c. Offset (in bytes) from the beginning of the file where to start reading and sending data<br>d. Output mode, including manual mode as described in DR2.6 and automatic modes as described in DR2.7<br>e. File read mode as described in DR2.8 | Partial | e is not complete. |
| DR2.5 | This module shall provide the following display and status | Full | |

| | | | |
|---|---|---|---|
| | information to the user:<br>a. The pathname of the file being transmitted<br>b. The number of blocks transmitted from the file<br>c. The current position in the file<br>d. The size of the file | | |
| DR2.6 | This module shall provide a manual output mode where each block of data is loaded from the disk and sent individually under user control. | Full | |
| DR2.7 | This module shall provide automatic output modes that include the capability to:<br>a. Send out the contents of a file once, several times, or continuously<br>b. Send out a subset of a file once, several times, or continuously<br>c. Send out the blocks in a file or subset of a file one or more times before sending out the next block | Full | |
| DR2.8 | This module shall provide file read modes that include the capability to:<br>a. Load consecutive blocks from a file based on a fixed offset<br>b. Load consecutive blocks from a file based on a synchronization pattern at the beginning of each block<br>c. Load consecutive blocks from a file based on a length field within each block<br>d. Load consecutive blocks from a file based on a header added by the log module | Partial | a, d only |
| DR3 | The SIMSS shall store and retrieve data to and from disk files on various storage devices. | Full | |
| DR3.1 | The SIMSS shall store and retrieve data to and from hard drives. | Full | |
| DR3.2 | The SIMSS shall store and retrieve data to and from CDs. | Full | |
| DR3.3 | The SIMSS shall store and retrieve data to and from Zip drives. | Full | |
| DR4 | The SIMSS shall store and retrieve data via in-line FTP. | | |

## 2.12 Modeling Requirements

| *MD* | *Modeling Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| MD1 | The SIMSS shall supply an interface to allow remote manipulation of internal data points. | Partial | SIMSS can now receive (name, value) data from Model |

| | | | |
|---|---|---|---|
| | | | Generator Prototype. |
| MD2 | The SIMSS shall provide the capability to model internal and telemetry parameters based on orbital position. | | |
| MD2.1 | This capability shall allow the user to set the following orbit parameters:<br>a. Time of orbit start<br>b. Orbit period<br>c. Eclipse duration time per orbit<br>d. Time from orbit start until eclipse | | |
| MD2.2 | This capability shall support the following modeling types:<br>a. Sine wave<br>b. Ramping<br>c. Exponential<br>d. Natural log<br>e. Polynomials up to the fifth order<br>f. Table-driven with interpolation<br>g. Table-driven without interpolation | | |
| MD2.3 | This capability shall be capable of using either raw or engineering units when modeling. | | |
| MD2.4 | This capability shall be capable of using the database to define:<br>a. Specific models (model type plus type-specific parameters)<br>b. Associations between models, parameters, and orbit status (day/night)<br>c. Granularity (how often to update based on model) in seconds | | |
| MD2.5 | This capability shall allow the user to see, enable or disable, or change any model read from the database. | | |
| MD2.6 | This capability shall allow the user to define additional models and associations. | | |
| MD2.7 | This capability shall be capable of displaying:<br>a. The current and next value of any modeled parameter (EU or raw)<br>b. Time until next value is applied<br>c. The model being used for a modeled parameter<br>d. A graphic plot of recent values for any modeled parameter | | |
| MD3 | The SIMSS shall provide the capability to model internal and telemetry parameters based on spacecraft events. | | |
| MD3.1 | Spacecraft events shall include:<br>a. Specific command received<br>b. Operator directive indicating that an event has occurred | | |

| | | | |
|---|---|---|---|
| | c. Telemetry or internal parameter going into a specific range<br>d. Telemetry or internal parameter going out of a specific range<br>e. Reaching a specific spacecraft time | | |
| MD3.2 | This capability shall support the following modeling types:<br>a. Sine wave<br>b. Ramping<br>c. Exponential<br>d. Natural log<br>e. Polynomials up to the fifth order<br>f. Table-driven with interpolation<br>g. Table-driven without interpolation | | |
| MD3.3 | This capability shall be capable of using either raw or engineering units when modeling. | | |
| MD3.4 | This capability shall be capable of using the database to define:<br>a. Specific models (model type plus type-specific parameters)<br>b. Associations between models, parameters, and events<br>c. Granularity (how often to update based on model) in seconds | | |
| MD3.5 | This capability shall allow the user to see, enable or disable, or change any model read from the database. | | |
| MD3.6 | This capability shall allow the user to define additional models and associations. | | |
| MD3.7 | This capability shall be capable of displaying:<br>a. The current and next value of any modeled parameter (EU or raw)<br>b. Time until next value is applied<br>c. The model being used for a modeled parameter<br>d. A graphic plot of recent values for any modeled parameter | | |
| MD4 | The SIMSS shall supply an interface to allow user or mission-specific extensions to model science instrument data. | | |
| MD5 | The SIMSS shall provide a module with the capability of reading a file containing module directives (a scenario file) and of passing that information to a module. | Full | |
| MD5.1 | This module shall be capable of reading a file one line at a time, extracting each line as a directive, and of passing the directive to another module. | Full | |
| MD5.2 | This module shall allow for lines in the file, in addition to directive lines, of the following types:<br>a. Comment, where the entire line is ignored<br>b. Sleep, where the execution of the file is paused for | Full | |

| | | | |
|---|---|---|---|
| | an amount of time supplied in the line<br>c. Start scenario, which would start a scenario based on a file pathname supplied in the line<br>d. Conditional (IF & While clauses) execution within scenario script files. | | |
| MD5.3 | This module shall be capable of sending directives to various modules based on information supplied in the directive line:<br>a. Regular set value directives<br>b. Set container item with simple expression<br>c. Set container item with other container items<br>d. Boolean expressions<br>e. With channel number (# num) specified at the beginning of the line. | Partial | a-d are implemented in the SIMSS Library. Unary negate operator may not work all the time.<br>If no channel number (# num) specified, the default is channel 1. |
| MD5.4 | This module shall be capable of accepting from an external module, pathnames of scenarios to execute. | Full | Special formatted directives shall be used at the module connected upstream. |
| MD5.5 | This module shall provide the user with the capability to indicate the files to read, up to a maximum of five files. | Full | |
| MD5.6 | This module shall provide the user with the capability to stop, start, or pause file execution at any time. | Full | Generated scenarios can only be stopped by project stop |
| MD5.7 | This module shall provide the user with status information including:<br>a. The name of the file being read<br>b. The current line number in the file<br>c. The contents of the current line in the file<br>d. Whether the module is running or stopped | Full | |
| MD5.8 | This module shall generate an event message for each directive line processed. | Full | Enabled by event message filtering from GUI |

## 2.12 Flight Dynamics Requirements

| *FD* | *Flight Dynamics Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| FD1 | The SIMSS shall provide an interface to support flight dynamics modeling. | | |
| FD1.1 | The SIMSS shall be capable of supporting mission specific attitude modeling. | | |
| FD1.2 | The SIMSS shall be capable of supporting mission specific orbit modeling. | | |

## 2.13 Spacecraft Simulation Requirements

| *SC* | *Spacecraft Simulation Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| SC1 | The SIMSS shall provide a generic, data-driven module with the capability to receive and validate commands, create and send telemetry, reflect commands received in telemetry, and support data and subsystem modeling. | | |
| SC1.1 | This module shall fulfill all of the requirements for telemetry generation listed under either TG1 or TG2. | | |
| SC1.2 | This module shall fulfill all of the requirements for command ingest listed under either CI1 or CI2. | | |
| SC1.3 | This module shall be capable of supporting all of the timing requirements listed under TM1, TM2, TM3, and TM4. | | |
| SC1.4 | This module shall be capable of supporting all of the modeling requirements listed under MD1, MD2, and MD3. | | |
| SC1.5 | This module shall be capable of supporting all of the flight dynamics modeling requirements listed under FD1. | | |
| SC1.6 | This module shall be capable of using command verification information from the database to reflect valid commands received in telemetry. | | |
| SC1.7 | This module shall be capable of executing a predefined model or script in response to a command received or other spacecraft event. | | |
| SC1.8 | This module shall provide the user with the capability to select the source and version of the database to use for module operations. | | |

# Section 3. Functional Requirements

## 3.1 SIMSS System-Level Performance Requirements

| *3.1* | *System-Level Performance Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| 3.1.1 | The SIMSS shall update status, data quality, and accounting information once every 10 seconds, at a minimum. | Full | For all modules implemented |
| 3.1.2 | The SIMSS shall acknowledge a request from a local user within 2 seconds of its entry. | Full | |
| 3.1.3 | The SIMSS shall start the execution of a local user request within 5 seconds of its entry. | Full | |
| 3.1.4 | The SIMSS shall be ready for operational use within 5 minutes of program execution exclusive of external dependencies. | Full | |

## 3.2 Serial Mode Performance Requirements

| *3.2* | *Serial Mode Performance Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| 3.2.1 | The SIMSS shall be capable of supporting up to four channels consisting of one command and up to three telemetry streams. | Full | ICS card: One command channel, two telemetry channels. Avtec card: One telemetry channel. Multiple channels can be achieved by using multiple cards. |
| 3.2.2 | The SIMSS shall be capable of receiving or transmitting three simultaneous fixed block length data streams at data rates from a minimum of 100 bits per sec (bps) to a maximum rate of 2 Mbps. | Full | Avtec card: 100 bps ~4 Mbps; ICS card: 900 bps ~2 Mbps. Maximum |

| | | | |
|---|---|---|---|
| | | | serial rate depends on size of frame. Rates are 2.0 Mbps and 1.54 Mbps, respectively, for 256-byte size frame. |
| 3.2.3 | The SIMSS shall provide the capability to receive or transmit a single variable block length bit stream at data rates up to 192 Kbps. | Partial | Transmit of variable-length blocks is not possible – a HW limitation for both ICS and Avtec cards. Need to package the variable-length blocks together and make it a fixed- length block. |

## 3.3 IP Mode Performance Requirements

| *3.3* | *IP Mode Performance Requirements* | *R5.0 Impl.* | *Comments* |
|---|---|---|---|
| 3.3.1 | The SIMSS shall be capable of supporting up to four channels consisting of one command and up to three telemetry streams. | Full | |
| 3.3.2 | The SIMSS shall be capable of receiving or transmitting three simultaneous fixed block length data streams at data rates from a minimum of 100 bits per sec (bps) to a maximum rate of 2 Mbps. | Partial | IP-Serial conversion limited the maximum data rate to 1 Mbps. Note that IP alone may have higher data rate. |
| 3.3.3 | The SIMSS shall provide the capability to receive or transmit | Partial | Can not |

| | | | |
|---|---|---|---|
| | a single variable block length bit stream at data rates up to 192 Kbps. | | handle variable frame length. |

# Abbreviations and Acronyms

| | |
|---|---|
| AOS | advanced orbiting systems |
| APC | automatic polarity check |
| APID | application ID |
| BI0-L | Bi-phase-Level |
| BI0-M | Bi-phase-Mark |
| BI0-S | Bi-phase-Space |
| bps | bits per second |
| CAB | Circuit Assurance Block |
| CCSDS | Consultative Committee for Space Data Systems |
| CD | compact disc |
| CLCW | command link control word |
| CLTU | command link transfer unit |
| CRC | cyclic redundancy check |
| CUC | CCSDS un-segmented time code |
| DGIB | DSN/GSFC Interface Blocks |
| DMSO | Defense Modeling and Simulation Office |
| DSN | Deep Space Network |
| EDOS | EOS Data Operations System |
| EOS | Earth Observation System |
| EU | engineering units |
| FARM | Frame Acceptance and Reporting Mechanism |
| FOT | flight operations team |
| FTP | File Transfer Protocol |
| GMT | Greenwich Mean Time |
| GSFC | Goddard Space Flight Center |
| GSTDN | Ground Space Tracking Data Network |
| GUI | graphical user interface |
| HLA | High Level Architecture |

| | |
|---|---|
| Hz | Hertz |
| ID | identification/identifier |
| IP | Internet Protocol |
| ISA | Industrial Standard Architecture |
| JSC | Johnson Space Center |
| Kbps | kilo bits per second |
| Mbps | mega bits per second |
| MDM | Multiplexer-Demultiplexer |
| MHz | mega Hertz |
| NASA | National Aeronautics and Space Administration |
| NASCOM | NASA Communications |
| NRZ-L | Non-Return-to-Zero-Level |
| NRZ-M | Non-Return-to-Zero-Mark |
| NRZ-S | Non-Return-to-Zero-Space |
| ODB | operations database |
| PCI | Peripheral Component Interconnect |
| PN | pseudorandom noise |
| POCC | Payload Operations Control Center |
| RTP | Real-Time Protocol |
| SCID | spacecraft ID |
| SIMSS | Scalable, Integrated, Multi-mission Simulation Suite |
| SMEX | Small Explorer |
| TCP | Transmission Control Protocol |
| TDM | time-division multiplexed |
| UDP | User Datagram Protocol |
| UTC | Universal Time Coordinated |
| UT1 | Universal Time 1 |
| VCDU | virtual channel data unit |
| VCID | virtual channel id |